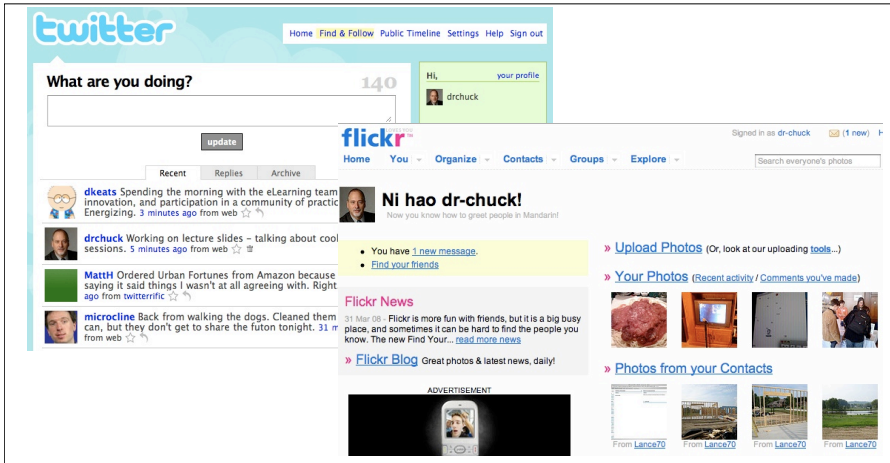
Cookies and Sessions
Maintaining State in HTTP

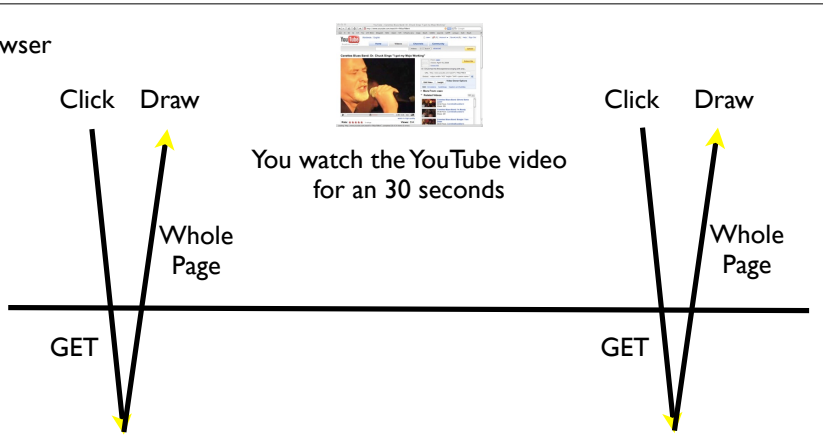
High Level Summary

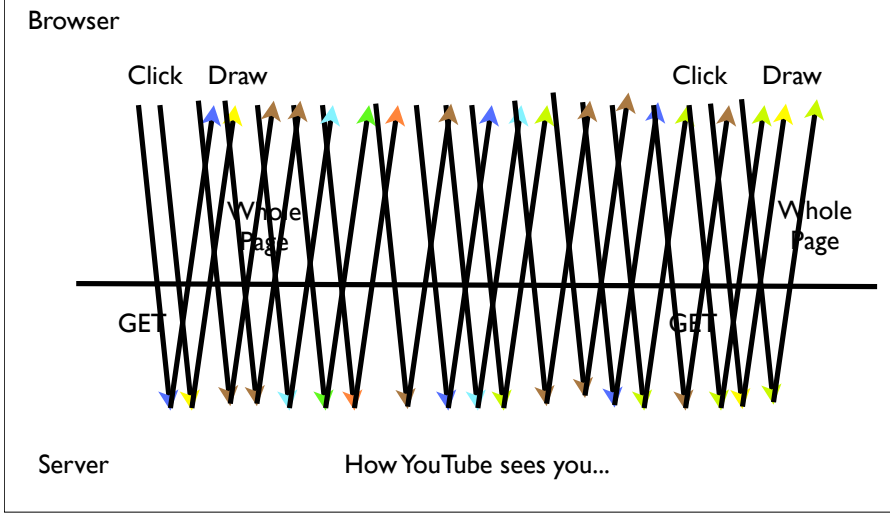
- The web is "stateless" - the browser does not maintain a connection to the server while you are looking at a page. Yu may never come back to the same server - or it may be a long time - or it may be one second later
- So we need a way for servers to know "which browser is this?"
  - In the browser state is stored in "Cookies"
  - In the server state is stored in "Sessions"

Some Web sites always seem to want to know who you are!



Other Web sites always seem to know who you are!





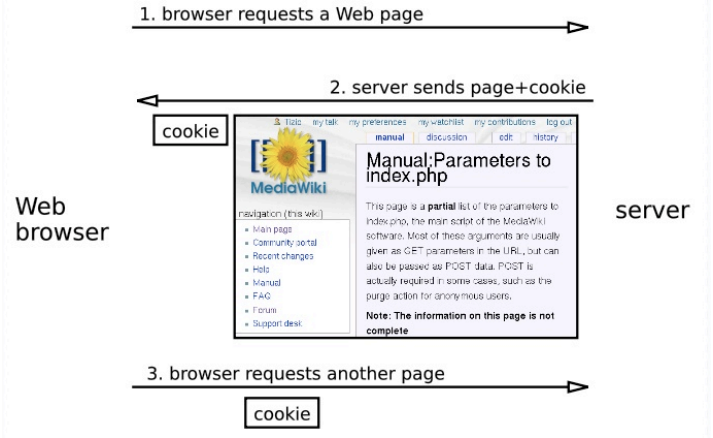
Multi-User

- When a server is interacting with many different browsers at the same time, the server needs to know *which* browser a particular request came from
- Request / Response initially was stateless - all browsers looked identical - this was really really bad and did not last very long at all.

Web Cookies to the Rescue

Technically, cookies are arbitrary pieces of data chosen by the Web server and sent to the browser. The browser returns them unchanged to the server, introducing a state (memory of previous events) into otherwise stateless HTTP transactions. Without cookies, each retrieval of a Web page or component of a Web page is an isolated event, mostly unrelated to all other views of the pages of the same site.

http://en.wikipedia.org/wiki/HTTP_cookie



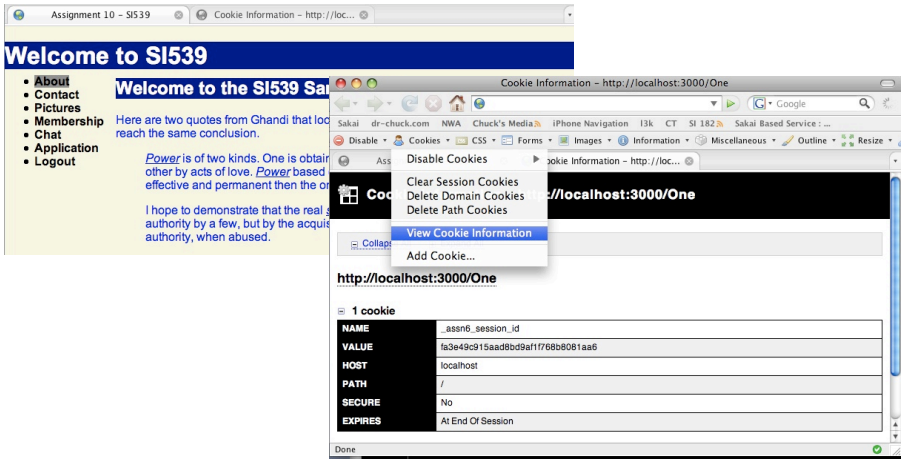
http://en.wikipedia.org/wiki/HTTP_cookie

Cookies In the Browser

- Cookies are marked as to the web addresses they come from - the browser only sends back cookies that were originally set by the same web server
- Cookies have an expiration date - some last for years - others are short-term and go away as soon as the browser is closed

Playing with Cookies

- Firefox Developer Plugin has a set of cookie features
- Other browsers have a way to view or change cookies



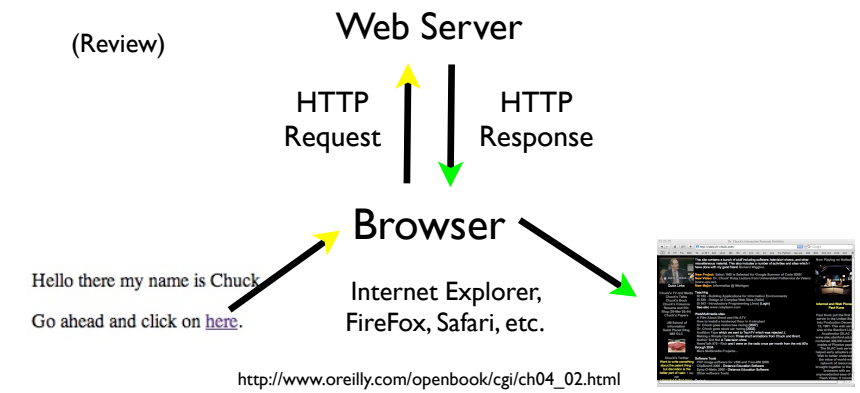
Cookies

- Identifying Individual Users
- The Web is "stateless"
- How do we make the web seem not to be stateless

Request Response Again!

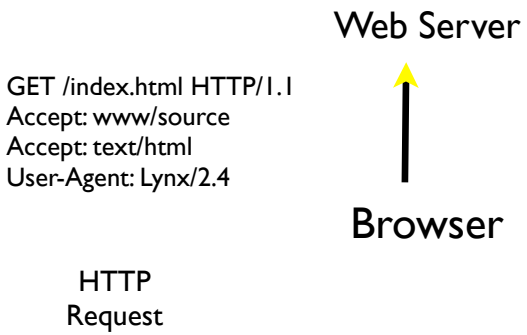
HTTP Request / Response Cycle

(Review)



http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle

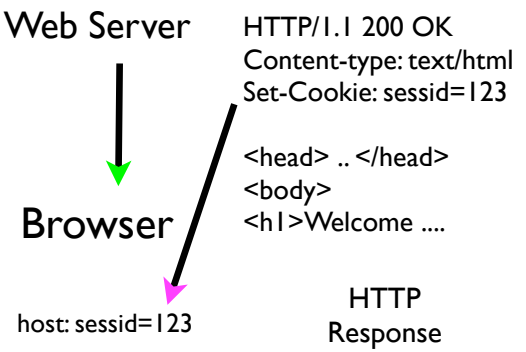


We do or initial GET to a server. The server checks to see if we have a cookie with a particular name set. Since this our first interaction, we have not cookies set for this host.

http://www.oreilly.com/openbook/cgi/ch04_02.html

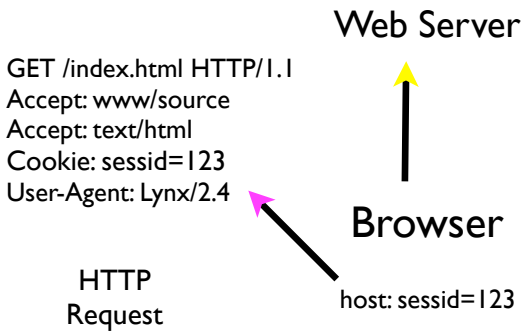
HTTP Request / Response Cycle

Along with the rest of the response, the server sets a cookie with some name (sessid) and sends it back along with the rest of the response.



http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle

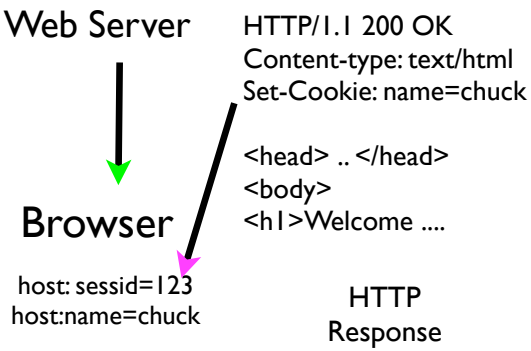


From that point forward, each time we send a GET or POST to the server, we include any cookies which were set by that host.

http://www.oreilly.com/openbook/cgi/ch04_02.html

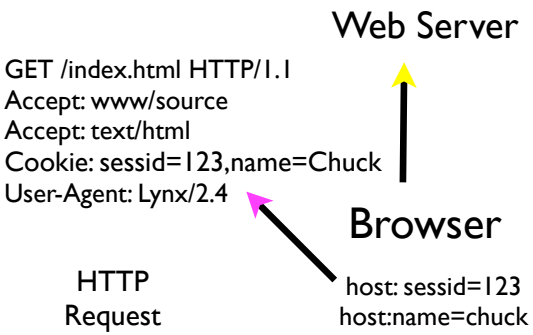
HTTP Request / Response Cycle

On each response, the server can change a cookie value or add another cookie.



http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Request / Response Cycle

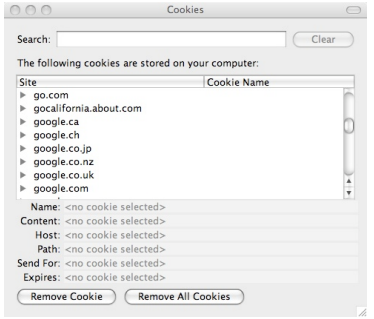


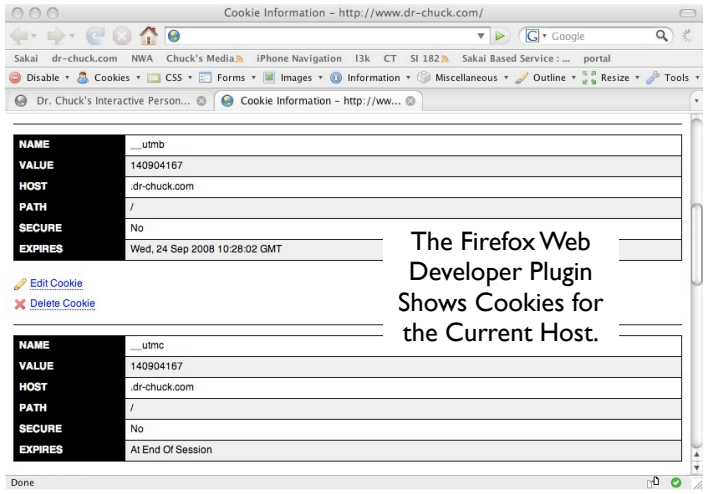
From that point forward, each time we send a GET or POST to the server, we include all the cookies which were set by that host.

http://www.oreilly.com/openbook/cgi/ch04_02.html

Security

- We ony send cookies back to the host that originally set the cookie
- The browser has *lots* of cookies for lots of hosts
- To ses all Cookies: Firefox -> Preferences -> Privacy -> Show Cookies

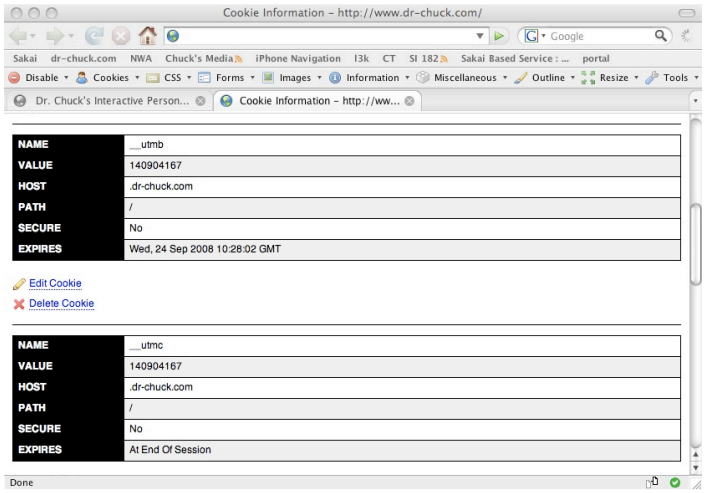




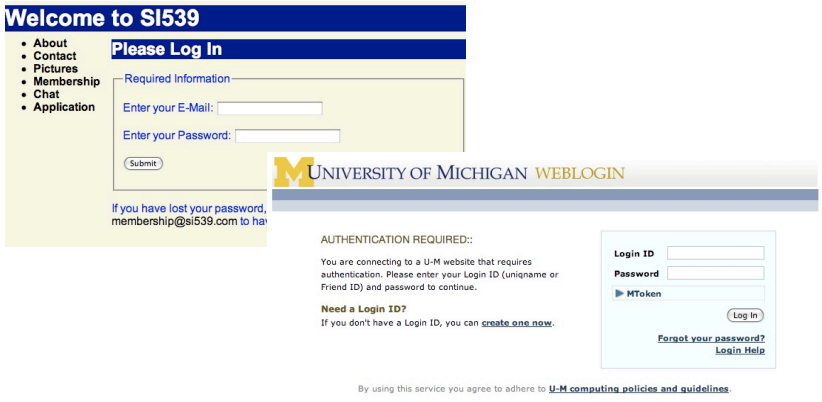
The Firefox Web Developer Plugin Shows Cookies for the Current Host.

Two Kinds of Cookies

- Two kinds of cookie
  - Long-lived - who you are - account name last access time - you can close and reopen your browser and it is still there
  - Temporary - used to identify your session - it goes away when you close the browser



Using Cookies to Support Sessions and Login / Logout



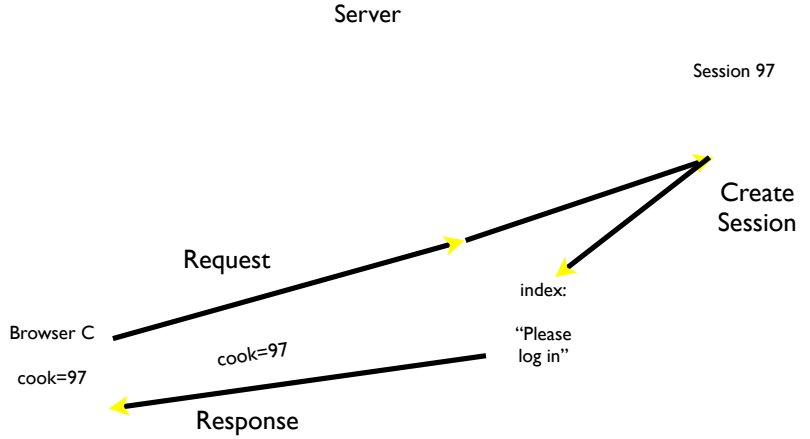
Some Web sites always seem to want to know who you are!

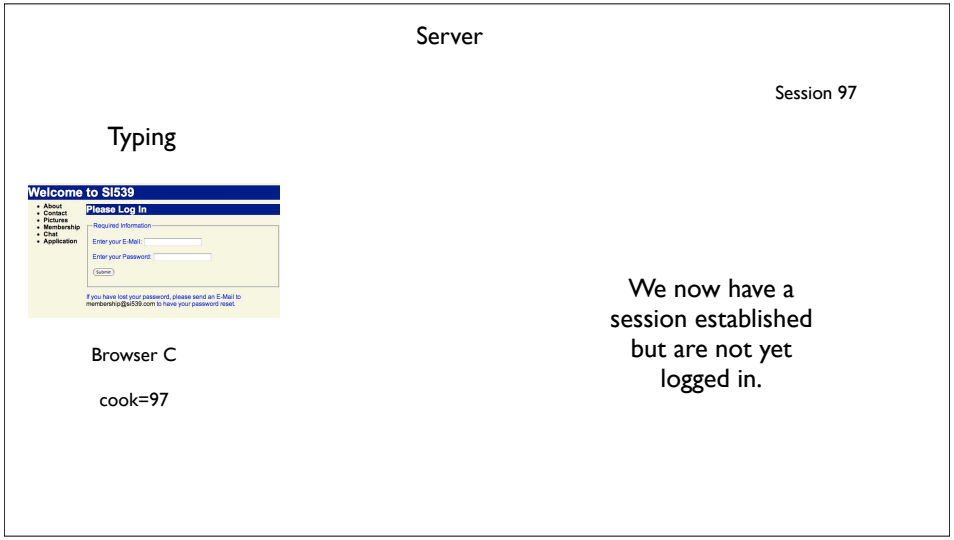
In The Server - Sessions

- In most server applications, as soon as we meet a new browser - we create a session
- We set a session cookie to be stored in the browser which indicates the session id in use
- The creation and destruction of sessions is generally handled by a web framework or some utility code that we just use to manage the sessions

Session Identifier

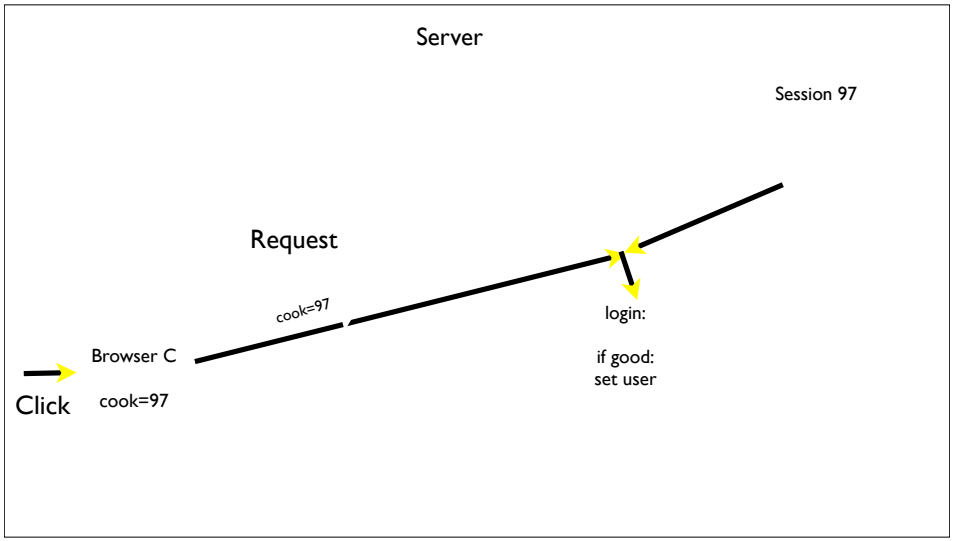
- A large, random number that we place in a browser cookie the first time we encounter a browser.
- This number is used to pick from the many sessions that the server has active at any one time.
- Server software stores data in the session which it wants to have from one request to another from the same browser.
  - Shopping cart or login information is stored in the session in the server

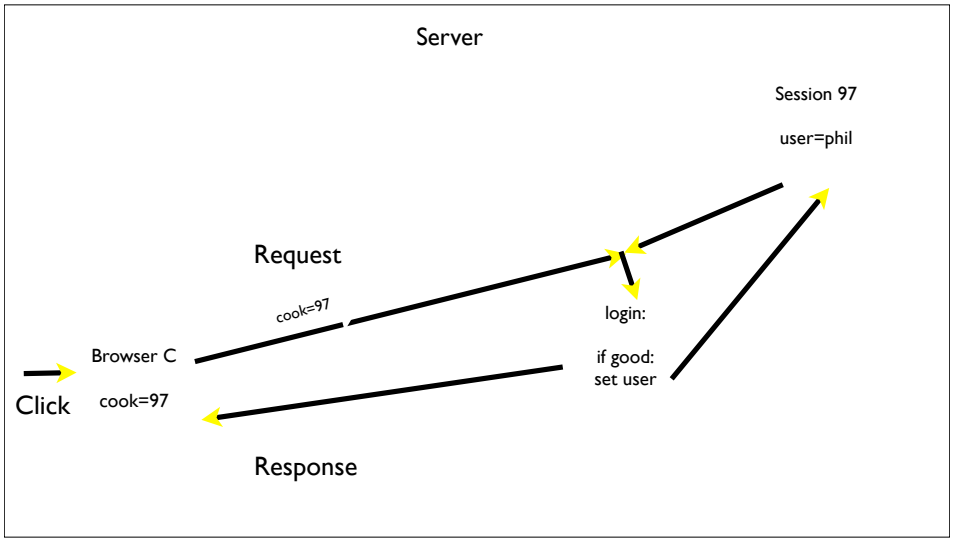


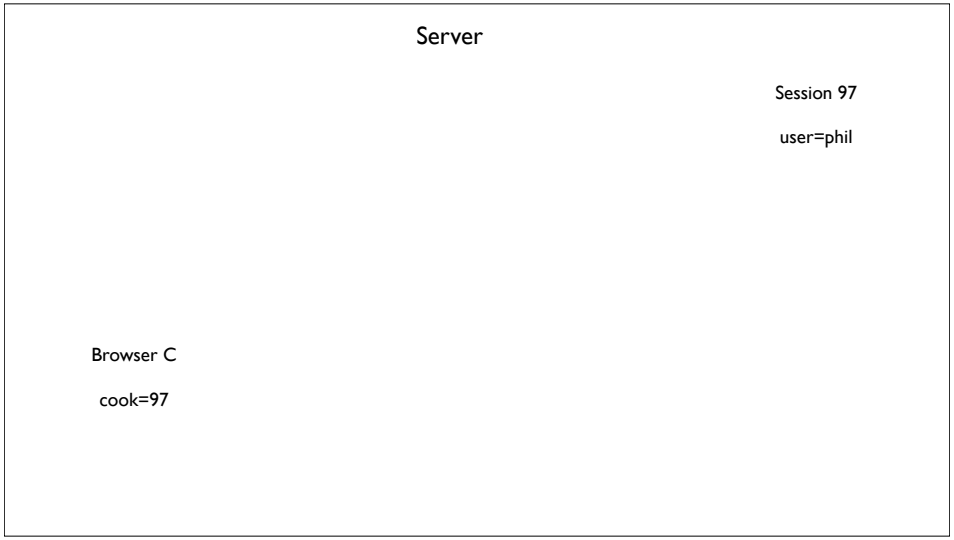


Login / Logout

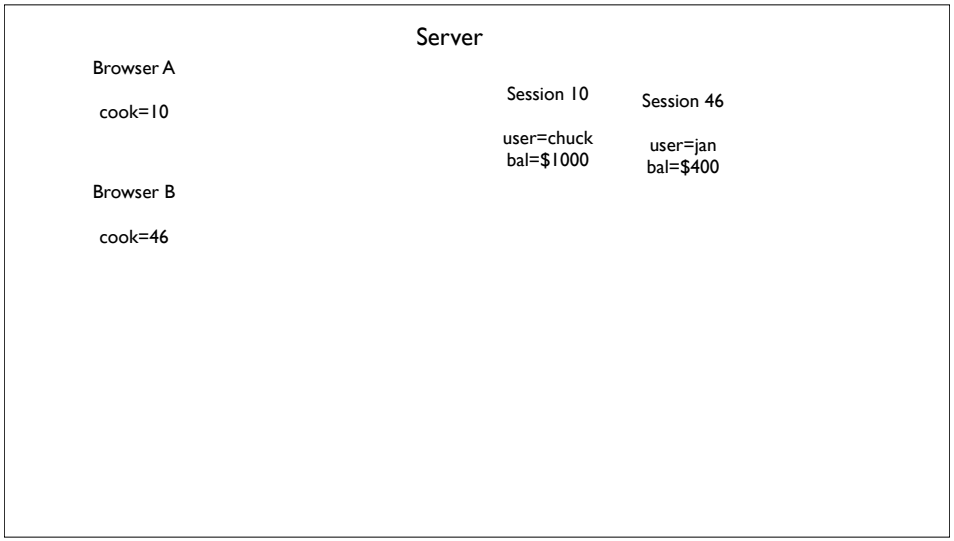
- Having a session is not the same as being logged in.
- Generally you have a session the instant you connect to a web site
- The Session ID cookie is set when the first page is delivered
- Login puts user information in the session (stored in the server)
- Logout removes user information from the session

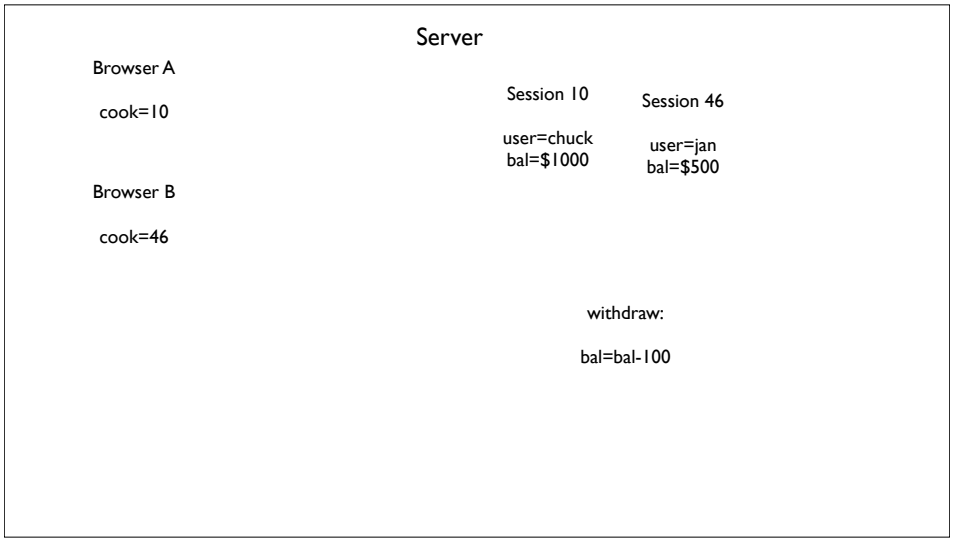


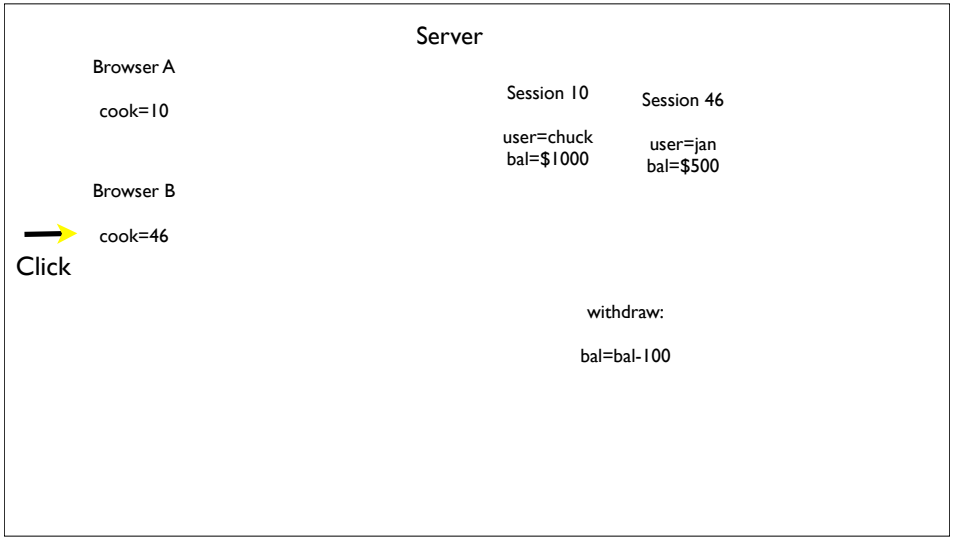


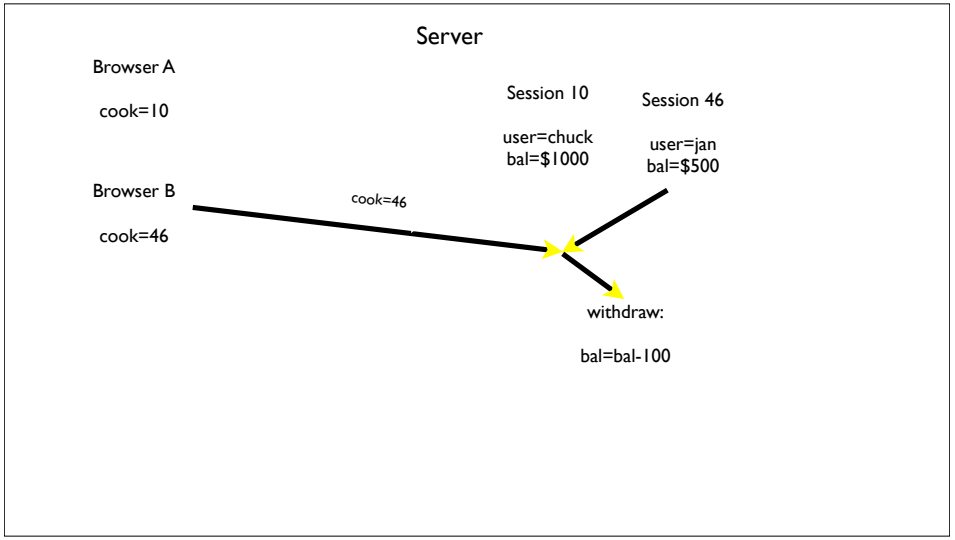


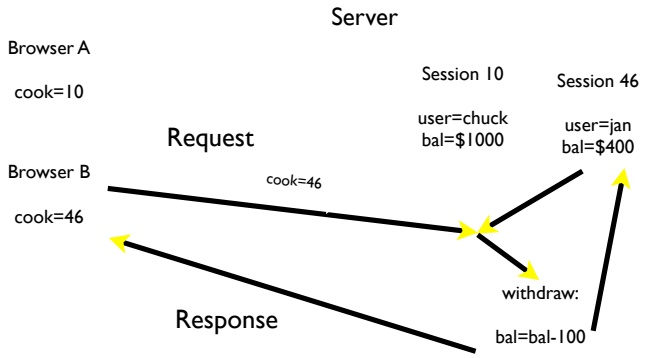
Using Sessions for Other Stuff


Server
Browser A
cook=10
Browser B
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$400


Server
Browser A
cook=10
Browser B
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100


Server
Browser A
cook=10
Browser B
cook=46
Click
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100


Server
Browser A
cook=10
Browser B
cook=46
cook=46
Session 10
user=chuck
bal=$1000
Session 46
user=jan
bal=$500
withdraw:
bal=bal-100

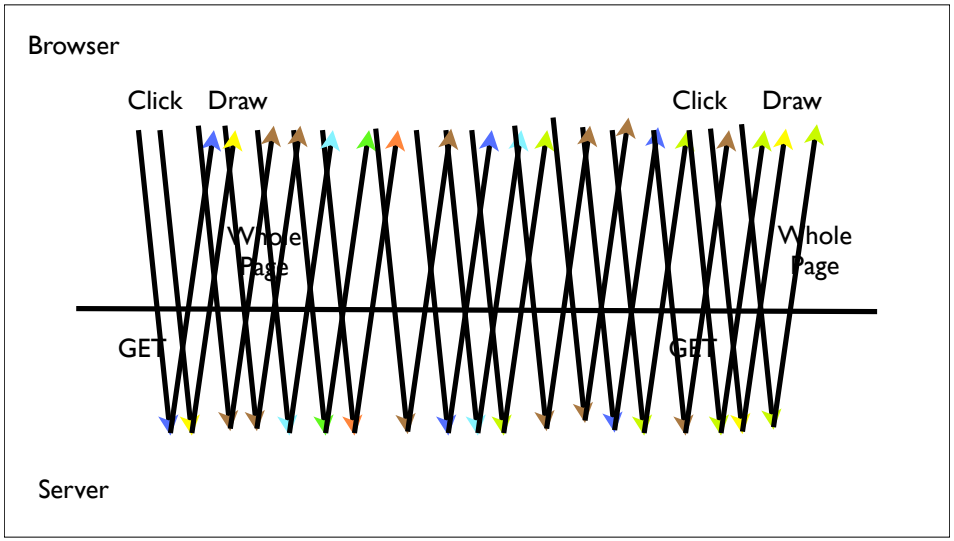


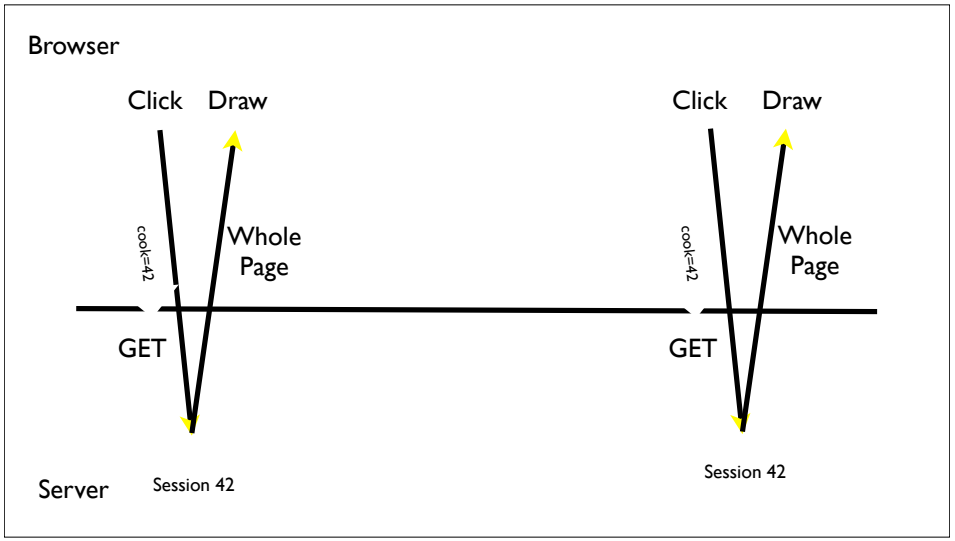
Review...

High Level Summary

- The web is "stateless" - the browser does not maintain a connection to the server while you are looking at a page. Yu may never come back to the same server - or it may be a long time - or it may be one second later
- So we need a way for servers to know "which browser is this?"
  - In the browser state is stored in "Cookies"
  - In the server state is stored in "Sessions"

How you see YouTube...





Cookie/Session Summary

- Cookies take the stateless web and allow servers to store small "breadcrumbs" in each browser.
- Session IDs are large random numbers stored in a cookie and used to maintain a session on the server for each of the browsers connecting to the server
- Server software stores sessions *somewhere* - each time a request comes back in, the right session is retrieved based on the cookie
- Server uses the session as a scratch space for little things